Language Resources and Evaluation (2005) 39: 45–64
DOI 10.1007/s10579-005-2695-2

Accuracy and Suitability: New Challenges for Evaluation

MARGARET KING
TIM/ISSCO, School of Translation and Interpretation, University of Geneva, Uni-Mail, 40 blvd du Pont d'Arve, CH-1205, Geneva
E-mail: Margaret.King@issco.unige.ch

1. Paradigms of Evaluation

In the early 1990s, when Antonio Zampolli was persuading the European Commission to launch the EAGLES initiatives, there were two prevailing paradigms in the evaluation of language technology software.

The first is the oldest. Since the 1950s, evaluations had been carried out on behalf of specific clients, who had mainly been interested in whether a particular system or a particular type of software met with the needs imposed by their own specific interests. In the field of language technology, a favourite object of such evaluations was machine translation. The ALPAC report, published in 1966 (ALPAC, 1966) was the earliest and the most notorious evaluation of this kind. Essentially, it looked at machine translation from the point of view of a government agency – probably an intelligence agency – asking whether machine translation could provide an economic alternative to the use of human translators. In finding an answer to the question, the evaluators did look at the results typically produced by systems of the day, though only as a primary factor contributing to the overall economic considerations. (This should not be taken as denigrating their work on evaluating results: their concern for experimental design and for identifying relevant metrics was laudable, unusual and very influential on much later work).

Later machine translation evaluations were often carried out on behalf of specific clients, who regarded the evaluation results themselves as being of commercial value. In these cases, neither the evaluation methods used nor the results were made publicly available.

Van Slype (1979), in a report commissioned by the European commission, gives an overview of those machine translation evaluation methodologies and techniques which were publicly accessible at the time, and Falkedal (1994) is a later attempt to synthesize experience to date.

Data base query is another application with a long history of evaluation. Woods (1973) describes informal field testing of the LUNAR system through monitoring the treatment of 110 queries during demonstration of the system, and Damerau (1980) reports more extensive field testing of TQA, a transformational grammar based front end linked to a pre-existing data base of town planning data, over a period of two years from late 1977 through 1979. The emphasis on field testing of data base query systems is also reflected in Jarke *et al.* (1985) and Whittaker and Walker (1989).

Implicit in these evaluations is the idea that different contexts of use impose different requirements: a good system in one context may be useless in another. There is a very simple demonstration of this in the case of machine translation systems; a system translating from French into English is of no possible use to someone who needs to translate from Danish into Greek. But of course the specific language pair is only one need among potentially very many. There are contexts where speed is of the utmost importance, contexts where the quality of the output translation outweighs any other consideration, contexts where human intervention can be contemplated and contexts where it is totally excluded; the list could grow almost indefinitely. In many cases, too, there is a trade off between needs: a very slow system may be acceptable if it produces good quality output; an otherwise acceptable system may be rejected because it will not accept documents in a particular format, or because it requires a particular kind of computer before it can be run. The need to take into account the fit between what a system can offer and what may be needed in a specific context of use became totally explicit with the publication of the JEIDA evaluation methodology for machine translation systems (JEIDA, 1992; Nomura and Isahara, 1992). The JEIDA methodology is based on constructing a graphical representation of a specific intended context of use, and a second graphical representation capturing the features a particular system has to offer. Overlaying one representation with the other allows the evaluator to see if there is a match between the two.

The second evaluation paradigm is that made familiar by evaluation campaigns, although it is foreshadowed by a proposal made in the context of progress and diagnostic evaluation by a group at Hewlett Packard (Flickinger *et al.*, 1987). They argue that although no evaluation tool could be developed for use with natural language processing systems in general, it should be possible and useful to develop a methodology for a single application domain (data base query) in a context where there are common assumptions. The main dimension considered relevant for evaluation of a generic system (i.e. a system not specifically tailored for use with one particular data base) is the functionality of the system. The criteria are linguistic and computational: the system should be able to treat a wide range of linguistic phenomena and should be able to generate the correct data base query from the natural language input.

The earliest evaluation campaigns were initiated by the Advanced Research Projects Agency (ARPA, sometimes also known as the Defence Advanced Research Projects Agency, DARPA, as it is at present). The declared aim of the campaigns is focused on advancing a core technology by encouraging competition amongst research teams working in that domain. Participants are typically expected to take part in regular conferences, where the results achieved by any one participating system are compared to the results of all other systems. The competition is conceived of as friendly rivalry, leading to participants gaining awareness of the most successful technological choices and perhaps seeking to incorporate them into their own work even at the price of abandoning some of their own ideas. There have been several such campaigns, some of them running over very long periods of time. Amongst the better known ones are the fact extraction campaigns realised through the Message Understanding Conferences (MUC, Grishman and Sundheim, 1996; Hirschman, 1998a, b) and the Text Retrieval Evaluation campaigns (TREC: TREC, 2005), which are still, after many years, enjoying increasing popularity. A rather less well known ARPA campaign was the machine translation campaign, singled out here because of its relevance to later discussion (White and O'Connell, 1994).

The (D)ARPA campaigns have inspired many others, too many to give an exhaustive list here. The domains covered are many and various, ranging over almost all sub-fields of human language technology from part of speech tagging and morphological analysis to word sense disambiguation. The TREC conference itself has expanded into a number of tracks, each dealing with a specific area in the general field of information and document retrieval. The TREC home page (http://trec.nist.gov/tracks.html) lists as TREC tracks all of cross-language retrieval, document filtering, retrieval in a specific domain (genomics), high accuracy retrieval, retrieval based on human interaction with retrieval systems, retrieval of new (i.e. not redundant) information, question answering, robust retrieval, video retrieval and retrieval over a data set which is a snapshot of the web, as well as a track concerned with scaling up evaluations to deal with much larger document collections than those used traditionally in TREC evaluations. The Cross Language Evaluation Forum (CLEF) campaigns (Peters, 2002), a spin-off from TREC, aim at promoting research into the design and development of user-friendly, multilingual and multi-modal retrieval systems, and also cover a very varied number of tracks (http.//www.clef-campaign.org). The interested reader will be able to find these and other recent campaigns easily enough by looking at the proceedings of the Language Resources and Evaluation (LREC) conferences, which since 1998 have provided a forum for the discussion of evaluation in the field of language technology. (Antonio Zampolli was one of the founders of LREC). Several of the earlier campaigns are discussed in (Sparck-Jones and Galliers, 1996).

What distinguishes the ARPA/DARPA campaigns from the sorts of evaluation described briefly at the beginning of this section is the focus of the evaluation. Only the functionality of the systems being evaluated is taken into account: context of use is held to be essentially irrelevant. To shift perspective a little in anticipation of later discussion, it is implicitly assumed that whatever the context of eventual use, the functionalities tested by the evaluation will play an indispensable role. (This is what the Hewlett Packard group meant, I think, by "shared assumptions"). Producing the best results (where best is, as we shall see later, defined by the evaluation itself) is held to be a strong indicator of superior underlying technology, and since the avowed aim of the campaign is to advance the core technology, producing the best results is interpreted as showing that pursuing the underlying technology is likely to prove productive.

Concentration on functionality alone also leads to a very strong emphasis on the definition of acceptable metrics. The campaigns by their nature compare different systems: they lose their rationale unless teams agree to participate willingly. If the metrics are perceived as being unfair, or biased towards some particular technological choice, enthusiasm for participation will diminish in consequence, and the campaign itself will thereby lose credibility. A great deal of important work on metrics which has been of direct use to the evaluation community in general has come out of the evaluation campaigns, especially since their organisers have actively encouraged criticism and discussion of the metrics used.

Taking the results achieved as the only indicator of the quality of the underlying technology also accounts for the black box philosophy of the evaluation campaigns. Black box evaluations only consider the outputs a system produces from a given set of inputs: there is no direct evaluation interest in looking at the internal workings of the system, as a glass box evaluation might. This allows systems with radically different underlying technologies to be directly compared, in conformity with the philosophy of the evaluation campaign itself. But it blocks consideration of some aspects of software which have been thought important elsewhere. For example, the only way in this paradigm to estimate a system's potential is to look at how its performance changes in practice over a period of time: there is no way of weighing up the potential fruitfulness of investment in system development as a preliminary to carrying out the investment.

2. Complementarities and Disparities between the Paradigms

For many people in the early and mid-1990s, the two paradigms sketched briefly above were perceived as being in radical opposition. Proponents of

functionality focused evaluation held that context-based evaluation was the domain of managers and administrators, and not of academic research workers who should restrict themselves to areas they understood. Proponents of context-based evaluation accused their opponents of living in ivory towers, and suggested that if they continued to neglect the needs of the real world they would find themselves lost in sterile discussion of systems nobody wanted and nobody would pay for.

The aim of the EAGLES evaluation working group (EAGLES, 1996) was to profit from both strands of previous evaluation work, pulling them together by designing a general framework for the evaluation of language technology systems from which the design of particular evaluations could be deduced. Although reconciliation of the two paradigms was not an explicit part of the mandate, the general framework in practice allows for reconciliation: an evaluation designed to take into account a specific context of use is an instantiation of the general framework, but so too is an evaluation designed as part of an evaluation campaign. Thus the apparent opposition disappears. The next section examines this claim in more detail.

3. EAGLES and ISO

Inspiration for the EAGLES approach was found in ISO work on standardisation, where it is a basic tenet that quality is always decided by reference to a user and his needs. The idea is that the user is always there, even when his presence is not explicit. The source of this idea when applied to the construction of software is fairly intuitive: it is quite hard to imagine anyone investing time and effort into the specification and creation of a piece of software unless he believes, consciously or unconsciously, that someone one day might find the software useful. If this argument is convincing, then even in functionality based evaluations a user is implicit in the sense that there is an assumption – albeit never overtly spelt out – that the functionalities around which the evaluation is constructed are exactly those functionalities needed by some otherwise unspecified community of users.

ISO published its first standard on the evaluation of software in 1991, just before the first EAGLES evaluation group officially started work (ISO 9126, 1991). On the grounds that a standard applying to the evaluation of software should a fortiori apply to the evaluation of language technology software, the EAGLES evaluation working group decided to use the ISO standard as the basis for its own work. The 1991 standard set out a quality model for software and offered pre-normative guidelines on how an evaluation might be designed and executed. The quality model is constructed on the basis of six quality characteristics: functionality, efficiency, usability, maintainability, reliability and portability. Not all quality characteristics are necessarily of the

same importance in any given specific evaluation. Their relative importance is decided by the evaluator on the basis of user needs, and is reflected in the evaluation design.

The quality model applies to the behaviour of the system when it is in operation. ISO work is based on the hypothesis that there is what one might call a quality chain. Internal quality is a property of the conception and coding of the system, and is measured by internal metrics. It predicts, at least partially, external quality, which is the quality of the system seen from outside, when it is running. External quality in its turn is at least a partial predictor of what ISO calls ''quality in use'', the quality of a system when it is being used by a user to accomplish some specific task. Quality in use can only really be evaluated *in situ*, in the user's own work context. The quality model therefore concentrates on external quality, on the grounds that there are generalizations that can usefully be made about what factors will enter into external quality, no matter what the type of software is, and that without satisfactory external quality, quality in use is very unlikely.

The notion of a quality chain in the software's life cycle was already present in embryonic form in the 1991 version of the standard to which we have been referring. Over the last few years, a new version of that standard has appeared in several parts. In these, the quality chain and the relations between the different kinds of quality have become totally explicit, and are discussed in considerable detail. (ISO 9126, 2001–2004)

If we now look at the two paradigms of evaluation in the light of the ISO quality model, we can see that context based evaluation makes explicit the needs of specific users in terms of all the quality characteristics. Evaluation campaigns deliberately neglect all the quality characteristic other than functionality, but the definition of desirable functionalities, as it is made explicit and concrete through the definition of metrics to be applied during the execution of the evaluation, can itself be projected onto a hypothetical user, a critical part of whose needs are held to be satisfied by any software with the requisite functionalities.

Both paradigms fit into the ISO model: in the context based evaluation paradigm, all of the quality characteristics are considered in determining and making explicit a user's needs. In the functionality based paradigm, the relative importance of all quality characteristics other than functionality is reduced to zero.

4. General Frameworks and Specific Needs

The ISO standards are meant to support the design of individual evaluations. In line with that, the 1991 9126 standard contained a set of pre-normative guidelines on how the process of evaluation should be designed, executed and

reported. These guidelines have subsequently become the topic of a new series of standards in the 14000 series. (ISO 14589, 1999–2001). Both ISO standards are meant to apply to any kind of software. They abstract away from the particular nature of the software in order to generalize at a level where generalization is possible.

EAGLES hoped to be able to provide specializations of the ISO quality model, which would flesh out the quality characteristics in terms of particular types of language technology software. The aim, for some given kind of software, was to provide a detailed instantiated quality model, such that an evaluator designing an evaluation for that type of software could pick out from the model just those parts which matched the user needs relevant to the particular evaluation. Thus, when the evaluator picked out that part of the quality model relevant to his particular case, he essentially acquired an evaluation scheme that was ready to use, modulo defining the relative importance of quality characteristics and sub-characteristics.

There is an obvious tension here: quality models are meant to support description of specific sets of user needs; how can they be used to model what it means for a particular kind of software to be of acceptable quality in all possible contexts – or, being realistic, at least in a significant number of different contexts? The EAGLES solution to this was to think in terms of classes of users, in much the same way that consumer magazines think in terms of classes of customers: they compare cars, for example, by looking at whether they would be a good car for someone with a large family, or for someone with strong ecological concerns, or for someone who travels a lot and so on. It should be possible to carry out the same sort of exercise for language technology systems, defining classes of users with similar needs. A quality model could then be constructed which reflected the needs of a given class of users: the general quality model would be the union of the quality models thus constructed.

Early exercises in executing this idea investigated the evaluation of grammar checkers and translation memory systems within EAGLES, and of spelling checkers within the TEMAA project (TEMAA, 1996). By far the most ambitious attempt to date, however, has been the construction of an evaluation framework for machine translation systems (FEMTI), carried out through a joint project (ISLE) of the European Union and the National Science Foundation of the United States of America, to which the Swiss Federal Office for Education and Science also contributed. FEMTI is a rather substantial piece of work which has involved considerable collaborative effort. There is no space to describe it in any detail here, but the interested reader is referred to (Hovy *et al.*, 2002a, b). Work on FEMTI is far from finished; however, its authors hope that even in its incomplete form it may prove useful, and have made it publicly available on the web at two mirrored sites: http://www.issco.unige.ch/projects/isle/femti. and http://www.isi.edu/natural-language/mteval.

The EAGLES framework has also been successfully applied to the design of specific evaluations of spelling checkers (Paggio and Underwood, 1998; Starlander and Popescu-Belis, 2002), of dialogue systems (Blasband, 1999) and of dictation systems (Cannelli *et al.*, 2000).

Work on machine translation evaluation was only one rather small part of the ISLE project: it will perhaps come as no surprise that once again Antonio Zampolli was one of the instigators of ISLE.

5. What is a User?

Discussion so far has been aimed at distinguishing two evaluation paradigms in terms of where they situate the definition of user needs, essentially claiming that the ISO quality model and the EAGLES/ISLE framework take into account user needs springing from the intended context of use of the software as well as the functionalities he might require, whereas evaluation campaigns define user needs (whether they intend to or not) solely in terms of the functionalities of the system. As we move towards identifying new challenges in evaluation of language technology software, we need to take a closer look at exactly how functionality represents user needs, but before we do so, it will be useful to spend some time emphasizing that users can come in all shapes and sizes, and are not necessarily end-users. A few examples will help to make this clear. As I sit here typing, I am a user of a text processing system and of the platform on which it sits. I am also an end-user. A university computing committee decided what text processor would be offered on university installations, and what the platform would be. They too can be thought of as users. I may be happy or unhappy with the software they decided to provide, they may be happy or unhappy with the consequences of their decision to make that provision. The text processor I am using calls on a spelling checker as one of its functionalities. The text processor is a user of the spelling checker. Whilst the text processor itself is unlikely to be happy or unhappy with the spelling checker it calls on, the manufacturer and the vendor of the text processing software, for their different reasons, may be satisfied or unsatisfied. The range of entities who may be users in the ISO and in the EAGLES sense is very large indeed. What makes them all users is that they have some task to perform, and that they propose to use the software being evaluated in the accomplishment of that task.

6. Functionality Revisited: Suitability and Accuracy

With this preliminary remark, we can return to the ISO 9126 quality characteristics, using now the most recent version of the standard. As mentioned above, the 1991 9126 standard has been replaced by two series of new

standards. These began to appear in 1998; different parts of the two series were published over the years between 1999 and 2004, and one document is still in preparation. (The bibliography gives details). The 9126 series now covers the quality model, internal metrics, external metrics and quality in use metrics. A new 14598 series is devoted to the process of evaluation, giving an overview, standards for planning and management and for the evaluation process from the points of view of developers, acquirers and evaluators. The part that concerns us the most here is the quality model, contained in part 1 of the 9126 series.

The earlier account of the 1991 quality model may have given the impression that the quality characteristics were monolithic entities. That was not true then and is not true now. Each quality characteristic is broken down into a number of sub-characteristics, and bottoms out in metrics which allow the performance of a system to be measured with respect to that sub-characteristic. The ISO standard legislates for only two hierarchical levels, quality characteristics and sub-characteristics. The EAGLES/ISLE formalisation of the quality model allows for as many hierarchical levels as are needed to achieve a level at which measurable attributes can be distinguished. This is primarily an extension motivated by practical considerations when working with specific types of software, and does not carry with it any change in theoretical stance: in particular, the first two levels of the hierarchy coincide with those legislated for in the ISO standard.

The functionality characteristic breaks down into suitability, accuracy, interoperability, security and compliance. For the purposes of this paper the last three will be ignored: definitions are given in the ISO standard, which the reader is urged to consult. In the 2001 version of part 1 of the 9126 standard, suitability is defined as "*the capability of the software to provide an appropriate set of functions for specified tasks and user objectives*". Accuracy is defined as "*the capability of the software product to provide the right or agreed results or effects*". It is important to notice that user needs only appear in the first of these definitions. This leads to an interpretation of accuracy as something very close to conformity to specifications: a software is accurate if it produces the results or effects that its specifications say it will. Suitability is intimately linked to user needs: it is decided by the task to be accomplished and the objectives of the user.

It should be said immediately that the interpretation of accuracy set out above is not something explicitly stated in the ISO standard. It does, however, match an intuitively satisfying distinction, and one which seems to run through ISO work.

We pointed out earlier that the evaluation campaigns involve black box evaluations: their specifications are therefore not open to inspection within the campaign. However, the results to be produced are, in a very strong sense, agreed results. This is especially clear in the case of those campaigns

which rely for their implementation on the creation of a set of data. A classic TREC evaluation, for example, relies on creating a collection of documents, a set of queries pertinent to those documents and a set of relevance judgements. A system's performance is judged on to what extent the documents retrieved in response to a given query match those pre-determined by the relevance judgements as being the documents which should be retrieved. Thus the system aims at producing a set of *agreed* results; what is at issue is the accuracy of the system as defined above. This notion of striving for agreed results is reinforced in those cases where part of the data is used to guide system development. Typically, in these cases, the data set is divided into two parts. One part, the training data, is made available to system developers. It serves both as a guide to what the system should strive to produce, and, in some cases, as basic material on which the system can be trained during development. The other part of the data set is used during the evaluation to test whether the developed system has in fact achieved the agreed results.

We can now restate the distinction we have been making between the two paradigms of evaluation. Context based evaluation spreads the definition of what has to be evaluated over all the quality characteristics. Functionality focused evaluation concentrates on one particular aspect of functionality, whether the system can produce the specified or agreed results.

A new question is beginning to emerge here; to what extent is it in general plausible that accuracy and suitability coincide? This question will occupy the next section, where we shall begin to see that the answer poses problems, although of rather different kinds, for both of the evaluation paradigms.

7. When do Suitability and Accuracy Coincide?

When software performs a well-defined task, it is perfectly natural to expect that suitability and accuracy will coincide: a program which calculates factorial (*n*) is useful to someone or some process that needs to calculate factorial (*n*), just as a program that orders items in a list alphabetically is useful to someone who needs to produce alphabetical lists. But coincidence of accuracy and suitability in the general case is a rather more subtle matter, especially as the sorts of tasks that we try to accomplish with the aid of software become more complex.

Terminology extraction programs offer a fairly intuitive example. One hypothesis about the nature of terms is that they are strings of words which tend to recur in a text or group of texts on the same technical subject. Commercial products have appeared on the market which use this hypothesis as the basis for specifying the behaviour of the terminology extraction tool. Typically, the user is asked to give a minimum length for the string of words

and a minimum number of times that the string will recur. Armed with these two parameters, the system produces from an input text a list of all those strings of words of specified length which recur the specified number of times. Early terminology extraction tools conformed exactly to these specifications, and were thus accurate in the ISO-derived technical sense we have been using. It takes little reflection though to realise that very few terminologists or translators found them of any practical use.

The problems, of course, are with the definition of a term and therefore with the specifications. Single words can be terms; if the minimum length of the string is stipulated as one word, we should simply get a list of all the word types appearing in the text, including far too many which are not terms. If we stipulate that the minimum length is two words, we shall miss all the single words, and still have a substantial number of strings like ''on the'' or ''given that'' which are not terms. Furthermore, with the definition as given, morphological variation will cause us to miss some two word sequences which are terms, for example ''extraction tool'' and ''extraction tools'' in the last paragraph. The point here, of course, is not to say that the specifications are extraordinarily naïve. It is rather that, in this case, accuracy in the sense of conformity to specifications and suitability in the sense of helping a user to achieve a task just simply do not coincide.

A rather similar argument can be made with the current generation of search engines. If (as is nearly always the case) a search engine is guided by key words formulated by the user, finding all the documents containing those key words means that the engine conforms to its specifications; it is accurate. But that does not mean that the results produced are actually going to help the user to achieve his task; we have all experienced the dreadful moment of being presented with millions of documents which may well contain what we have been looking for, although we know we shall never find it. It is for this reason of course that many search engines try to order the documents found by some criterion of relevancy: they are trying to improve the suitability of the results.

A new element is creeping in here, and one which contributes greatly to the appearance of new challenges. Search engines (and in a lesser way the sort of terminology extraction tools described above) depend intimately on the user who interacts with them. Some humans are better at web searching than others. Some basic skills of web searching can be taught, such as how to formulate a boolean request, how to limit search to the title of the page or to the URL and so on. But even equally armed with basic skills, some people find what they are searching for more quickly and more easily than others: there is a talent involved, as well as training and experience. And no piece of software can be faulted for failing to take into account the impoverished talents of the human who tells it what to look for.

New and more complex kinds of software are beginning to appear which render the problem sketched above even more acute. Data mining software, for example, searches for patterns in the data submitted to it, presenting the results to the user perhaps as a set of associations between different elements in the data.[1] All its specifications can do is to define how associations are to be found: they can do nothing to specify what counts as an interesting or a useful association. That depends in part on the data (bad data will produce valid but bad associations), but also, and critically, on the human interacting with the software, whose job it is to look at the associations initially produced and to guide the software by telling it to ignore those variables which lead it astray. In other words, it is part of the intended functioning of the software that a human user will have an intimate influence on whether or not suitable results are produced – the software is not working alone.

This is the first of our new challenges. There are well known and well accepted ways of evaluating the accuracy of data mining software, for example looking at whether association rules are correctly discovered and formulated or at whether the clusters formed by a clustering algorithm are internally coherent but sufficiently distinct one from another. But how can we find ways to evaluate this new kind of suitability, when suitability depends not only on the accuracy of the software but also on the talent of the user?

The challenge can only become more acute as software becomes more complex and more ambitious. Both paradigms of evaluation face the challenge, as witnessed by (Hawking *et al.*, 1999) in the functionality focused paradigm and by work in the Parmenides project (Spiliopoulou *et al.*, 2004; Vasilakopoulos *et al.*, 2004) in the context based paradigm. But the difficulty of meeting the challenge gains additional poignancy in the user-oriented paradigm, with its much vaunted insistence on taking the user fully into account.

8. The Importance of Having the Right Answer

Another way of looking at the challenge would be to regard it as a problem in finding an appropriate set of metrics for suitability in cases where it makes little sense to measure system performance independently of human performance. In a way, this is a familiar problem, and one that we might again relate to the increase over time in the complexity of the tasks to be accomplished with the aid of software. We might even be able to distinguish a continuum of complexity directly related to the difficulty of establishing metrics.

With rather simple tasks, it is usually the case that we know what the right answer should be. Earlier we mentioned tasks like calculating factorial or ordering a list alphabetically. An early example taken from language

technology might be transcription of spoken words: providing that we know what word was spoken, it is very easy to check whether it was correctly transcribed, and thus to devise a metric based on what percentage of a reasonably sized sample of words is dealt with correctly. The very elegance of metrics like this is seductive: most readers will recognise the word error rate metric familiar from the evaluation of speech recognition systems, and some will be aware of recent efforts to adapt the metric to other areas like machine translation.

However, it is not always so easy to know what the right answer is. Complications arise whenever any sort of human judgement is involved. Thus, the classical TREC metric mentioned earlier refers to human judgement in defining the set of relevance judgements. There will of course be clear cases, where all or nearly all judges will agree that a document is relevant or is irrelevant. But there will equally certainly be a grey area, where judges do not agree.

The same applies to evaluations like the MUC evaluations. The task here is to extract from text the information needed to fill in slots in a template structure which represents the essential information contained in a stretch of text. There are two parts to the definition of the agreed results: the templates themselves and what the slot fillers should be. Again, there will be cases where the vast majority of human judges agree, and perhaps in this context rather more cases where humans do not entirely agree.

The communities involved in the TREC and MUC evaluations have tended to get round the problem of defining the required results by seeking consensus amongst those defining the evaluation and those participating in it. It has not always been easy to establish the consensus, but in the end, a working agreement has always been reached, and the evaluation has been able to proceed. The metrics used by TREC are again seductive in their simplicity: once the consensus solution to defining what counts or does not count as a relevant document has been accepted, it becomes easy to measure in terms of how many of the relevant documents in the document set are effectively found (the recall of the system) and of how many documents are wrongly (because irrelevant) retrieved by the system (its precision). Once again, many readers will be familiar with modifications and extensions of the recall and precision metrics to suit them to other applications of language technology.

I have talked about the word error rate, recall and precision metrics as being seductive just because of the very many attempts to adapt them to other applications. That some of those attempts are ill-founded is another topic for another paper. The main point here is to emphasize that a shift has been made from using the ''right'' results to specify a metric for accuracy to using a set of ''agreed'' results. And once what the right results are depends on a consensus agreement, we are on the way to no longer being able to

predict with any certainty that accuracy and suitability will coincide. In some quite perceptible sense, the boundary between acceptability and suitability has begun to blur.

Working with consensus and agreed results nonetheless does at least allow the definition of metrics that do not require human intervention in their application: human judgement is limited to definition of the agreed results and does not directly operate on the results produced by the system being evaluated.

Some applications by their nature preclude the definition of a right or agreed answer. One such is machine translation. Given any single sentence of reasonable length, it is rather unlikely that two human translators would come up with the same translation for it, yet both their translations may be equally acceptable. It is no accident then that machine translation furnishes us with several examples of metrics that try to avoid the issue of defining a right answer by evaluating the quality of output indirectly, relating it to some other criterion which can more easily be measured. One such metric asks subjects to complete a comprehension test after reading the translated text, on the grounds that the better the translation, the easier it will be to get the answers to the comprehension test right. Another, in splendidly ISO style, relates quality of the output directly to the ability to accomplish some task, asking whether, given that output, a human can sort documents into categories or produce a gist of the document's content (White and Taylor, 1998). Many of these metrics are ingenious and many have provoked controversy. But what concerns us here is to notice that human participation in their application is inextricable. For the first time, we are faced with the issue of separating out evaluation of the system from evaluation of the human. It is exactly for this reason that there has been so much recent work on trying to find machine translation metrics that, whilst accepting that definition of a single right or agreed answer is impossible, nonetheless eliminate the need for human participation in the metric. (Papineni *et al.*, 2001).

The final point on the continuum takes us to where accuracy becomes almost totally subordinate to suitability, the point reached in our last section. Here the difficulty shifts heavily in the direction of defining suitability, simply because we can no longer think in terms of reaching a consensus or, in EAGLES terms, thinking in terms of classes of users. Relating quality of output to the ability to carry out a specified task, as has been done in evaluating machine translation systems (see earlier discussion) and in some other task based evaluations (for example Hand, 1997) does not really offer a way out, since it involves being able to distinguish a (small) number of generic tasks which many users would like to be able to accomplish. In other words it is just another way of defining classes of users.

Let us use as an illustration of the problem the different users involved in the Parmenides project (cited above), a text mining system looking for patterns in

large quantities of text rather than in structured data. There are four sets of needs expressed by the users. One is to discover from a collection of archived material whether certain topics have already been discussed. A second is to pick out from a mass of material emerging trends in consumer behaviour. A third is to identify patterns of events leading to useful commercial intelligence, and the final need is to search very amorphous material in order to identify patterns that might help in signalling planned terrorist attacks. At a very general level of abstraction there is a common need: finding patterns in a mass of material. But the particular needs almost certainly cannot be satisfied by software which is generic. Critically, the functioning of the text mining software itself has to take into account the kind of information being looked for: to relate the problem to a technique we have already encountered, if the software starts by trying to extract facts and fill in templates, the nature of the templates will be different as a reflection of each set of user needs, and the rules which allow the facts to be extracted will also be significantly different.

What is happening here is that just as the dividing line between accuracy and suitability began to dissolve as agreed results replaced the right results as the centre of evaluation interest, now suitability has drowned out accuracy to the point where the borderline between external quality and quality in use is beginning to grow dim. In all these cases, it is at least theoretically possible to create a system tailored to a particular set of user needs and subsequently to find out whether it really supports a user in his task of gaining new insights or identifying new trends. But to do so removes almost entirely the main point of at least evaluation in the user-oriented paradigm. Carrying out the development needed to suit a text mining or a data mining system to a particular set of user needs requires a considerable investment, not only in hand-crafting the system but in preparing and cleaning the data or the text from which the data will be gleaned. In these circumstances, the main point of evaluating external quality of the system is to provide a basis for deciding whether the investment should be made. If external quality no longer predicts quality in use, the evaluation is no longer valid and loses its raison d'être. King and Underwood (2004) contains some discussion of these and related issues.

This, then, is our second new challenge: can we find a way to preserve the link between external quality and quality in use in the case of systems whose nature and complexity is such that developing the system and providing it with adequate and appropriate data on which to work requires already taking individual user needs into account?

9. The Shifting Sands of Moving Data

Earlier talk of web searching has already hinted at the final challenge to be signalled here. The last section pointed out that finding a right answer does

not only involve being able to agree on what the right answer might be, it involves there being a right answer to find. There, it was the nature of the application that in some cases precluded there being a right answer. In other cases, the nature of the data poses analogous problems. The information available on the web is inherently unstable: what is there today may be gone tomorrow. It is also available in vast quantities, so vast that no human could hope to be able to master it even to the extent of producing a list of all the information sources present. This means that although it is at least theoretically possible to check through a list of documents retrieved by a web search and decide whether or not they actually meet the criteria specified for the search, it is not even theoretically possible to check for silence: there is no way of knowing whether a relevant document has been missed.

The difficulty is only compounded if we consider the possibility of carrying out text mining over web documents. It has already been pointed out that the results of data or of text mining can be radically perverted by poor or inconsistent data. Much of the information available on the web is eccentric, unreliable or perverse. A system looking for patterns in it may come up with some very untrustworthy results.

Then too the amorphous nature of web information adds to the problem. Even at its best, a system which can only look, for example, at HTML documents and even inside that constraint cannot deal adequately with documents with different structures will be severely limited in the sort of results it can come up with.

This final challenge is perhaps the worst. Both of our previous challenges concerned, in different ways, difficulties raised by the collaboration of men and machines in trying to accomplish specific tasks. Here we are faced with a question of a quite different order: how can we assess how much confidence we can place in results gained by processing vast amounts of amorphous data, where the data itself is constantly changing and where we know some of its sources to be fundamentally unreliable?

10. Conclusion

A meditation on how user needs are represented in different kinds of evaluation has led us to the definition of three new challenges in evaluation theory. The first of these concerns evaluation when humans are inextricably involved in working with a computer system to produce results: how can we separate evaluation of the system from evaluation of the human? The second concerns the problem of defining user needs at a sufficiently generic level: how can we avoid having to deal with every user as a separate and individual case? The third concerns the nature of the data current and future systems are called upon to deal with: how can we separate out evaluation of the system from evaluation of the data?

These are, I believe, difficult challenges, and challenges of which we are only beginning to be fully aware. The stakes, though, are very high. The astonishing development of the web over the last 20 years has meant that vast amounts of information have become available at the click of a mouse. But wittingly and unwittingly, we have created a jungle; the problem now is not to access information but to manage and control it. Computers can process vast amounts of information in ways that people cannot, and in so doing they hold out the possibility of not just managing information but of using it creatively, stimulating the discovery of new connections between disparate elements, the formulation of new ideas stimulated by those new connections and ways of validating new ideas founded on human intuition. If we can find the ingenuity and inventiveness to meet the challenges identified here, we can look forward to a new era in the fertility of human thought. If we cannot, there is a very strong risk that we shall simply drown in a morass of unreliable, inconsistent and ultimately unusable information.

Acknowledgements

Most of the work on which this article is based has been collaborative work and has involved very large numbers of people. In particular, I should thank my colleagues from the two EAGLES projects and from the associated TEMAA project as well as those from the ISLE project and the Parmenides project. They are too many to mention by name. Andrei Popescu-Belis, Nancy Underwood and Agnes Lisowska have been partners in an on-going conversation about evaluation from which I have learnt much. I am of course solely responsible for any misrepresentations or poor arguments.

Note

[1] There are other techniques for data mining. However, what is said here about the search for association rules can be applied, mutatis mutandis, to those other techniques.

References

ALPAC. (1966) *Languages and Machines: Computers in Translation and Linguistics*. Report of the Automatic Language Processing Advisory Committee, Division of Behavioral Sciences, National Academy of Sciences, National Research Council Publication 1416, Washington, DC.

Ankherst M. (2001) *Human Involvement and Interactivity of the Next Generation's Data Mining Tools*. Workshop on Research Issues in Data Mining and Knowledge Discovery, Data Mining and Knowledge Discovery (DMKD) 2001.

AMTA. (1992) MT Evaluation: Basis for Future Directions. In *Proceedings of a workshop held in San Diego*, CA. Technical report, Association for Machine Translation in the Americas.

Blair D.C. (2002) *Some Thoughts on the Reported Results of TREC*. Information Processing and Management, 38/3, Pergamon Press, Tarrytown, NY, pp. 445–451.

Blasband M. (1999) Practice of Validation: The ARISE Application of the Eagles Framework. In *Proceedings of the European Evaluation of Language Systems Workshop*. Hoevelaken, Holland.

Canelli M., Grasso D., King M. (2000) *Methods and Metrics for the Evaluation of Dictation Systems: a case study*. LREC 2000, Athens, pp. 1325–1331.

Damerau F.J. (1980) *The Transformational Question Answering System: Description, Operating Experience and Implications*. Report RC8287, IBM Thomas Watson Research Center, Yorktown Heights, NY.

Doyon J., Taylor K., White J.S. (1998) The DARPA MT Evaluation Methodology: Past and Present. In *Proceedings of the AMTA Conference*, Philadelphia, PA.

EAGLES. (1996) *EAGLES Evaluation of Natural Language Processing Systems*. Final Report, EAGLES Evaluation Working group. Report EAG-EWG-PR.2 (ISBN 87-90708-00-8), Center for Sprogteknologi, Copenhagen.

Falkedal K. (1994) *Evaluation Methods for Machine Translation Systems: An Historical Overview and a Critical Account*. Internal report, ISSCO. Available from ISSCO.

Flickinger D., Nerbonne J., Sag I., Wasow T. (1987) *Towards Evaluation of NLP Systems*. Report, Hewlett Packard Laboratories, Palo Alto, CA.

Grishman R. (1997) *Information Extraction: Techniques and Challenges*. In Pazienza M.-T. (ed.), *Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology*, SCIE-97, Frascatti, Italy, pp. 10–26.

Grishman R., Sundheim B. (1996) *Message Understanding Conference-6: A Brief History*. Coling-96.

Hand T.F. (1997) *A Proposal for Task-based Evaluation of Text Summarization Systems*. ACL/EACL workshop on Intelligent Scaleable Text Summarization, Madrid, pp. 31–37.

Hawking D., Carswell N., Thistlewaite P., Harman D. (1999) Results and Challenges in Web Search Evaluation. In *Proceedings of the Eighth International Conference on World Wide Web*, Elsevier.

Hirschman L. (1998a) *Language Understanding Evaluations: Lessons learned from MUC and ATIS*. LREC-98, Granada, Spain.

Hirschman L. (1998b) The Evolution of Evaluation: Lessons from the Message Understanding Conferences. *Computer Speech and Language*, 12, pp. 281–305.

Hovy E.H., King M., Popescu-Belis A. (2002a) Principles of Context-Based Machine Translation Evaluation. *Machine Translation*, 16, pp. 1–33.

Hovy E.H., King M., Popescu-Belis A. (2002b) *Computer-Aided Specification of Quality Models for Machine Translation Evaluation*. LREC-02, pp. 729–753.

ISO/IEC 9126-1. (2001) *Software Engineering – Product Quality – Part 1: Quality Model*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC DTR 9126-2. (2003a) *Software Engineering – Product Quality – Part 2: External Metrics*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC CD TR 9126-3. (2003b) *Software Engineering – Product Quality – Part 3: Internal Metrics*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC CD 9126-4. (2004) *Software Engineering – Product Quality – Part 4: Quality in use Metrics*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC CD 9126-30. (in preparation) *Software Engineering – Software Product Quality Requirements and Evaluation – Part 30: Quality Metrics* – Metrics reference model and

guide. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 14598-1. (1999) *Information Technology – Software Product Evaluation – Part 1: General Overview*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 14598-2. (2000a) – *Software Engineering – Product Evaluation – Part 2: Planning and Management*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 14598-3. (2000b) – *Software Engineering – Product Evaluation – Part 3: Process for Developers*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 14598-4. (2000c) – *Software Engineering – Product Evaluation – Part 4: Process for Acquirers*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 14598-5. (1998) *Information Technology – Software Product Evaluation – Part 5: Process for Evaluators*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 14598-6. (2001) – *Software Engineering – Product Evaluation – Part 6: Documentation of Evaluation Modules*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

ISO/IEC 9126. (1991) *Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for their Use*. Geneva, International Organization for Standardization and International Electrotechnical Commission.

Jarke M., Turner J.A., Stohr E.A., Vassiliou Y., White N.H., Michielsen K. (1985) A Field Evaluation of Natural Language for Data Retrieval. *IEEE Transactions on Software Engineering, SE-11*, 1, pp. 97–113.

JEIDA. (1992) *JEIDA Methodology and Criteria on Machine Translation Evaluation*. Report, Japan Electronic Development Association, Tokyo.

King M., Underwood N. (2004) User Oriented Evaluation of Knowledge Discovery Systems. In *Proceedings of a Workshop at LREC-04*.

Minker W. (2002) Overview on Recent Activities in Speech Understanding and Dialogue Systems Evaluation. *International Conference on Speech and Language Processing (ICSLP)*, Denver, USA.

Nomura H., Isahara J. (1992) The *JEIDA Report on MT Evaluation. Workshop on MT Evaluation: Basis for Future Directions*. Association for Machine Translation in the Americas (AMTA), San Diego, CA.

Paggio P., Underwood N. (1998) Validating the TEMAA Evaluation Methodology: A Case Study on Danish Spelling Checkers. Natural Language Engineering 4/3, pp. 211–228.

Papineni K., Roukos S., Ward T., Zhu W.-J. (2001) *BLEU: A Method for Automatic Evaluation of MT. Research Report*, Computer Science RC22176 (W0109-022), IBM Research Division, T.J. Watson Research Center.

Peters C. (2002) *The Contribution of Evaluation: The CLEF Experience*. Special Interest Group in Information Retrieval (SIGIR), 2002.

Sparck Jones K. (2001) Automatic Language and Information Processing: Rethinking Evaluation. In Natural Language Engineering, 7(1), pp. 29–46.

Sparck-Jones K., Galliers J.R. (1996) *Evaluating Natural Language Processing Systems: An Analysis and Review*. Lecture Notes in Artificial Intelligence 1083, Springer-Verlag, Berlin/New York.

Spiliopoulou M., Rinaldi F., Black W.J., Zarri G.P., Mueller R.M., Brunzel M. Theodoulidis B., Orphanos G., Hess M., Dowdall J., McNaught J., King M., Persidis A., Bernard L.

(2004) *Coupling Information Extraction and Data Mining for Ontology Learning in Parmenides*. RIAO 2004, Avignon.

Starlander M., Popescu-Belis A. (2002) *Corpus-Based Evaluation of a French Spelling and Grammar Checker*. LREC-02, Las Palmas de Gran Canaria, Spain. pp.262–274.

TEMAA. (1996) *TEMAA Final Report*. Technical report LRE-62-070 (March 1996), Center for Sprogteknologi, Copenhagen, Denmark.

TREC. (2005) *Text Retrieval Conference (TREC) TREC-9 Proceedings*. Available from http://trec.nist.gov.

VanSlype G. (1979) *Critical Study of Methods for Evaluating the Quality of MT*. Technical Report BR 19142, European Commission, Directorate for General Scientific and Technical Information Management (DG XIII) Available from www.issco.unige.ch/projects/isle.

Vasilakopoulos A., Bersani M., Black B. (2004) *A Suite of Tools for Marking Up Textual Data for Temporal Text Mining Scenarios*. LREC-04, Lisbon.

Voorhees E.M. (2003) *Evaluating the Evaluation: A Case Study Using the TREC 2002 Question Answering Track*. HLT-NAAL.

Voorhees E.M. (2000) *The Evaluation of Question-Answering Systems: Lessons Learned from the TREC QA Track*. LREC-2000, Athens.

White J.S., O'Connell T.A. (1994) The DARPA MT Evaluation Methodologies: Evolution, Lessons and Future Approaches. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas (AMTA-94)*. Columbia, Maryland.

White J.S., Taylor K.B. (1998) *A Task-Oriented Evaluation Metric for Machine Translation*, LREC-98.

Woods W.A. (1973) *Progress in NLU – An Application to Lunar Geology*. AFIPS 42, pp. 441–450.

Whittaker S., Walker M. (1989) *Comparing Two User-Oriented Database Query Languages: A Field Study*. Technical Report HPL-ISC-89-060, Hewlett Packard Laboratories, Bristol, UK.

Yeh A.S., Hirschman L., Morgan A.A. (2003) Evaluation of Text Data Mining for Data Base Curation: Lessons Learned from the KDD Challenge Cup. *Bioinformatics*, 19 (Suppl. 1), pp. i331–i339.